Semantic body parts segmentation for quadrupedal animals

Semantic Body Parts Segmentation for Quadrupedal Animals

H. Haggag, A. Abobakr, M. Hossny, and S. Nahavandi

Centre for Intelligent Systems Research
Deakin University, Australia

***Abstract*—Although marker-less human pose estimation and tracking is important in various systems, nowadays many applications tend to detect animals while performing a certain task. These applications are multidisciplinary including robotics, computer vision, safety, and animal healthcare. The appearance of RGB-D sensors such as Microsoft Kinect and its successful applications in tracking and recognition made this area of research more active, especially with their affordable price. In this paper, a data synthesis approach for generating realistic and highly varied animals corpus is presented. The generated dataset is used to learn a machine learning model to semantically segment animal body parts. In the proposed framework, foreground extraction is applied to segment the animal, dense representations are obtained using the depth comparison feature extractor (DCF) and used for training a supervised random decision forest (RDF). An accurate pixel-wise classification of the parts will allow accurate joints localisation and hence pose estimation. Our approach records classification accuracy of 93% in identifying the different body parts of an animal using RGB-D images.**

***Index Terms*—RGB-D cameras, pose estimation, semantic segmentation and random decision forest.**

I. Introduction

Articulated objects pose estimation receives much interest from the computer vision community due to its wide range of practical applications. Joints localisation via semantic body parts segmentation, also known as pixel-wise classification, is an effective approach to marker-less pose estimation and tracking for human beings and animals. In this approach, every body pixel is assigned the discrete categorical label of the body part it belongs to, and subsequently interpolating body joint positions. Pixel-wise classification is more challenging than traditional object detection and recognition vision tasks. Foreground extraction and obtaining discriminative pixel level representations are the main contributory factors for such difficulty.

However, the advancements in depth technologies such as Microsoft Kinect have relaxed this task. The Kinect sensor was introduced in 2010 as an Xbox 360 external gaming peripheral. This RGB-D camera technology is patented by PrimeSense. The Microsoft Kinect allows people to play and interact with their console without carrying any controllers or requiring pose calibration. Nowadays, RGB-D cameras are used in areas far beyond gaming. The size [1], price [2] and accuracy [3]–[5] of these RGB-D cameras make it extensively used in many other research areas such as work safety and ergonomic assessments [4], [6], biometrics [7], [8], simulated crowd interactions [9], human machine interaction [10], [11] and 3D scanning [12], [13].

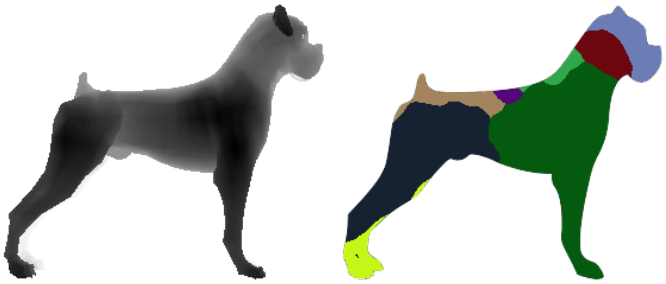
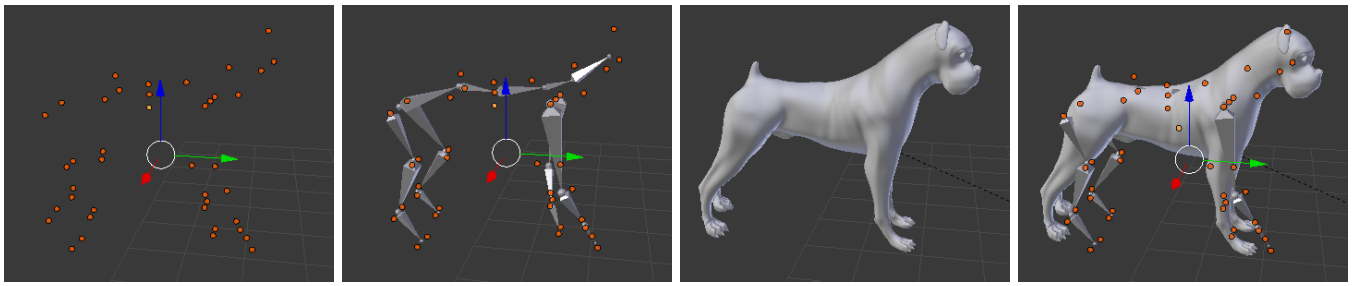


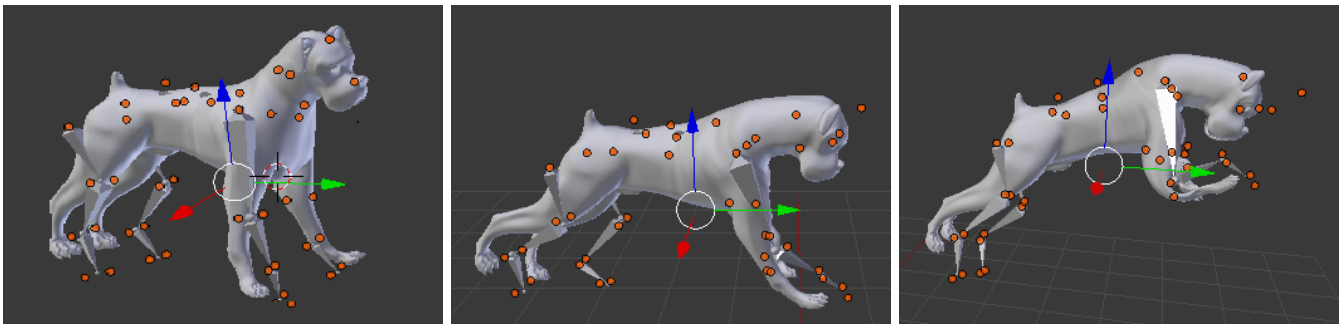
Fig. 1: Method overview: a) a synthetic depth image of a dog, b) shows a sample classification output of the RDF where each pixel is assigned the discrete categorical label of the body part it belongs to.

The most well-known approach in labelling and tracking is the one proposed by Shotton et al. [14]. Pixel classification in depth images was done using random decision forests (RDF) [15]–[17], trained on the pixel-wise representations obtained by the depth comparison feature extractor (DCF). Buys et al. [18] utilised both the colour and depth data to adapt on-line to new environments. In this approach, a kinematic model was used to construct the skeleton. Both of these approaches [14], [18] focus only on human pose estimation, however, animal pose estimation is also important for application domains like animal surveillance [19], safety [20], medical care [21] and gaming. The main limitation is the availability of fully labelled animals training dataset as it is difficult to instruct and record animal activities in front of a depth camera. Therefore, building on top of Shotton' [14] and Koen's [18] approaches, we start with building a diverse, synthesised and fully labelled animals training dataset. A set of virtual animal models with different anthropometric characteristics is designed. Animal activities such as running and jumping are recorded using a motion capture system. Then, the recorded motion data is mapped onto animal models. Finally, pixel labelling and rendering processes are applied. Fig. 1, shows a pair of synthesised depth and ground truth training images. The forest is trained on this dataset using discriminative and computationally efficient depth comparison features (DCF). Both RDF and DCF could be evaluated in parallel on a



(a) (b) (c) (d)

Fig. 2: The steps of modelling and animating a quadrupedal animal, a) shows the configuration of markers placed on animal's body, b) the traced animal skeleton using a motion capture system, c) displays the virtual model created using a 3D modelling software, and d) demonstrates the fitting process.



(a) (b) (c)

Fig. 3: Samples of animated frames for a dog walking, running, and jumping.

GPU [14], [16] allowing real time performance.

This paper is organised as follows: Section II describes our data generation approach. Feature extraction and RDF training are detailed in Section III. Section IV shows the experimental results and discussion. Finally, we conclude in Section V.

II. Data Generation

A marker-based motion-capture (MoCap) system is used to capture different animations of a dog such as walking and jumping. While a 3D modelling and computer graphics software such as Blender [22] is used to create different animal models. Then, a skeleton of a dog is constructed with different bones that are assigned to follow the marker trajectories. Finally, the created virtual models are fitted on the constructed skeleton. The process of modelling and animating the dog is depicted in Fig. 2. Samples of different captured animal activities are shown in Fig. 3.

At this stage the captured motion data accompanied with the constructed skeleton are mapped onto the dog model. The next step is annotation, a one-time manual process into which each body pixel in every depth frame is assigned the label of the candidate body part it belongs to. Simply, each vertex in the 3D model will be added to one vertex group, which represents a dog body-part. Subsequently, each of these vertex groups will be given a unique discrete categorical label. We currently discriminate between 17 dog body parts (one for the head, three for the front left leg, three for the front right leg, three for the back left leg, and three for the back right leg).

The training data generated previously is used to train the forest. Two steps are involved, first one is optimising a set of discriminative feature extractors. Secondly, training the randomised forest of trees using the MapReduce paradigm [23].

III. Random Decision Forests

In the first step, the state of the art simple depth comparison feature (DCF) is employed. The idea of the DCF had been first introduced in [24] as the difference of RGB pixel intensities. The DCF is used for characterising depth information due to its efficient computations while maintaining high discriminating signal and depth translation invariance [14], [18]. For every body pixel p in a depth image D, a feature vector of length 2000 is computed as the depth difference between this pixel p and another 2000 randomly sampled offset pairs as illustrated by the following formula:

$$f(p,\Theta) = d_D\left(p + \frac{o_1}{d_D(p)}\right) - d_D\left(p + \frac{o_2}{d_D(p)}\right) \quad (1)$$

where $d_D(p)$ is the depth value of pixel p, and $\Theta = (o_1, o_2)$ is a randomly sampled offset pair. The offset normalisation factor $\frac{1}{d_D(p)}$ is applied to ensure depth invariance feature response.

The storage cost of the dense DCF representations obtained per image is approximately 4 MB.

A. RDF Training

Randomised Decision Forests (RDF) [25] is an ensemble of decision tree predictors, each consisting of split and leaf nodes [26]. RDF has been proven fast and efficient for handling different data analysis problems such as classification, regression, clustering and dimensionality reduction in [27] and can be implemented efficiently on the GPU [16]. Each tree t in a forest of size N combines its posterior distribution over the pixel label $P_t(c|v)$ with all the other trees to produce the forest estimate $P(c|v)$. The following formula is for the used average ensemble model

$$P(c|v) = \frac{1}{N_t} \sum_{t=1}^{N_t} P_t(c|v) \tag{2}$$

where c is a candidate body part label, v is the feature vector extracted for a body pixel in the given depth image, and N_t is the number of trees in the forest. Background pixels are removed in a foreground segmentation pre-processing step.

As mentioned previously, the storage cost per image is approximately 4 MB. Consequently, as the training samples increase, the computational resources become a challenge. We used the horizontal scaling approach to tackle this problem. Being a candidate problem for key-value pair formulation, opened the door to use the MapReduce component of the Apache Hadoop distributed computing ecosystem [28]. A Hadoop cluster totalling 18 cores, 80 GB RAM, 6 TB storage is deployed for RDF optimisation.

Training a decision tree is mainly an optimisation of the split nodes parameters $\Phi = (\Theta, \tau)$, where $\Theta = (o_1, o_2)$ is a 2D offset pair and τ is a feature threshold. Therefore, to train a single tree, the following steps are performed. First, a random set of splitting candidates Φ is generated. Secondly, at each splitting node, the information gain is used as a criterion to select the best splitting candidate from the set Φ and binary divide the incoming training samples. This procedure is decomposed into a set of simple Map and Reduce tasks.

Using the MapReduce component of Hadoop is advantageous. First, Hadoop balances the computational load over a commodity of hardware. Secondly, horizontal scalability via adding extra nodes is valuable. Most importantly, training the RDF is computationally expensive. Therefore, the fault tolerance capabilities for Hadoop are tremendously beneficial. Another thing that makes MapReduce and Hadoop an adequate solution, is that the training process can be performed on the cloud using the IBM Compute SoftLayer that publicly supports a Hadoop MapReduce service.

The result of the forest learning process is a set of binary random decision trees (N). There are several hyper-parameters that affect the generalisation capabilities of the forest. However, tree depth has the most significant effect on the model performance [14]. Optimising a tree depth is an experimental process. In the following section, we contribute with results obtained using different number of levels.

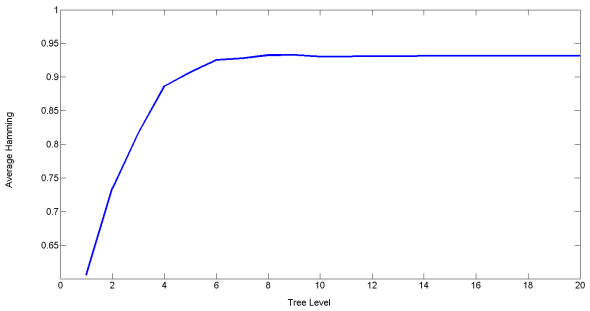


Fig. 4: The effect of varying number of tree levels over the average hamming accuracy.

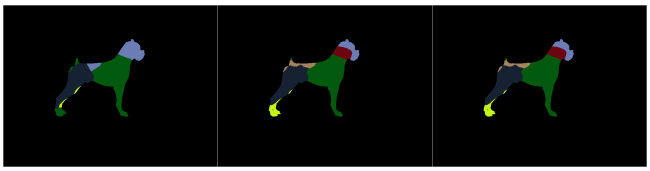
One strict requirement for our system is to ensure real time performance. Therefore, pixels evaluation is implemented on the graphics processing unit (GPU) [16]. An input test sample v is pushed simultaneously into all trees starting from the first split node until reaching its leaf. At each split node of a tree, a binary test is applied to compare the DCF outcome of the input sample against an optimised threshold. As soon as reaching a leaf node, the tree t outputs the conditional probability distribution $P_t(c|v)$, where c is the discrete label of the body part. Then, the produced tree posteriors are aggregated to constitute the forest estimate $P(c|v)$ of unseen test pixel v as shown in Eq. 2.

IV. Experimental Results

We have trained a random decision forest of size 4 trees to depth 20 using a synthetic and fully labelled training dataset of $9K$ images to discriminate between 17 discrete classes representing predefined dog body parts. While generalisation capabilities are affected by different model parameters, including forest size and the size of the training dataset, tree depth has the most significant influence on the overall model performance [14], [27]. Therefore, in this section, we describe the used performance metric and study the effect of varying the tree depth parameter on the generalisation behaviour of the model. The forest size is set to 4 and number of training images to $9K$.

A. Evaluation Criterion

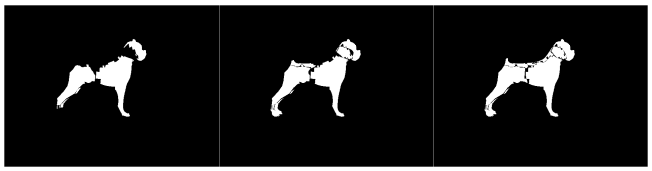
The trained forest is used to perform pixel-wise classification of input depth images generating fully labelled images. The hamming distance [29] is used to measure the accuracy of the resulting output images. Each pixel in the resulted image is checked against the same pixel in the ground truth image producing a new hamming accuracy map, also known as similarity map. Each pixel in the accuracy map will have the value 1 if the same pixel in both the reference and resulted images had the same label, and 0 otherwise. The final hamming accuracy is the sum of the ones of the similarity map divided by the total number of body pixels.



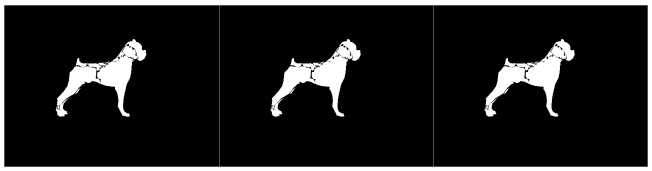
(a) Tree-2 (b) Tree-4 (c) Tree-6

(d) Tree-10 (e) Tree-15 (f) Tree-20

Fig. 5: Resulted labelled dog images using RDF trees that are trained to 6 different levels.



(a) Tree-2 0.73438 (b) Tree-4 0.88637 (c) Tree-6 0.92521



(d) Tree-10 0.93023 (e) Tree-15 0.93154 (f) Tree-20 0.93154

Fig. 6: Resulted similarity images using average hamming accuracy of RDF trees that are trained on a dog when using trees with 6 different levels. Body pixels of zero value indicates a misclassified sample.

TABLE I
The Effect of Tree Depth on the Real Time Performance

Trees Depth	Frames Per Second
2	140
4	120
6	110
10	100
20	40

Using deeper trees provide better generalisation performance while adding extra real time computational cost.

B. Trees Depth

To study the effect of the depth of the trees, the proposed method is evaluated using fixed forest size of 4 trees and varying number of levels $2, 4, 6, 10, 15$ and 20 respectively. Fig. 4, describes the effect of varying tree depth on the average hamming accuracy. Results demonstrates that deeper trees have better generalisation performance. However, slight gain has been achieved with trees deeper than 6 and overfitting begins after depth 10. Using larger training sets could help in avoiding the overfitting problem [14] and achieving better results from deeper trees. On the other hand, using deeper trees adds extra computational cost during model evaluation. Table I, describes the relationship between the number of levels and the number of frames per second. Fig. 5, shows the resulted labelled images when using the RDF trees with levels $2, 4, 6, 10, 15$ and 20. Fig. 6, shows the resulted similarity images of each scenario accompanied with the hamming accuracy.

The confusion matrix was constructed, where pixels of each dog body-part of the reference image were checked in the resulted image to indicate to which labels of other dog body-parts these pixels were assigned and understand the confusing body parts as shown in Fig. 7.

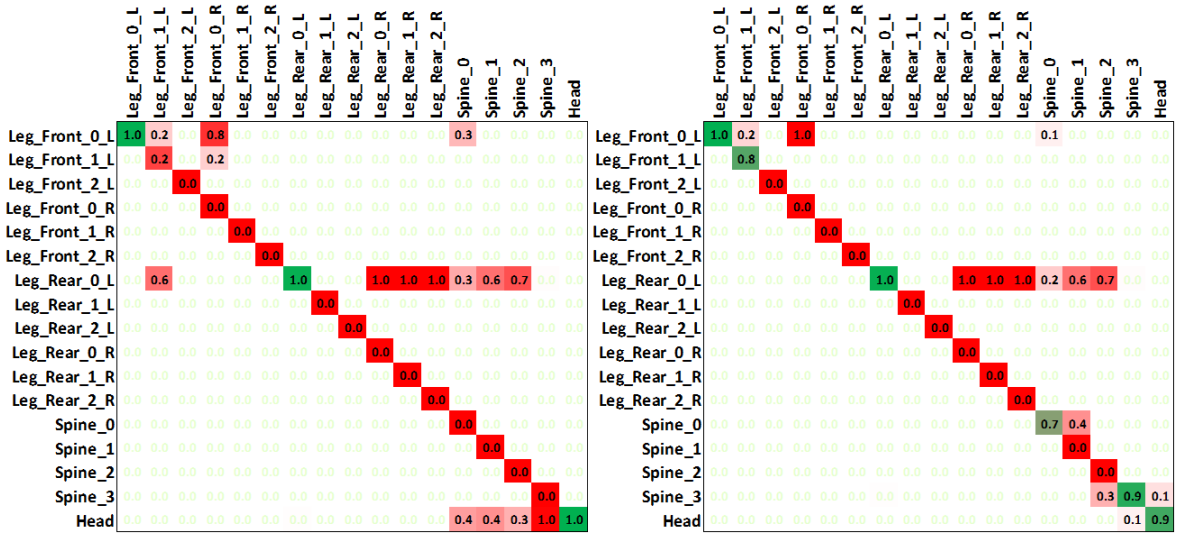
C. Discussion

Fig. 5, clearly demonstrates the failure of labelling and not being able to identify most of the dog body-parts when using low level trees, and better labelling when using higher level trees. To measure the accuracy, hamming distance was used to construct the similarity maps as shown in Fig. 6. The hamming accuracy between the reference images and the resulted images was measured, where only the matching pixels that have the same label in both the reference and resulted images are counted. Fig. 6, shows that the accuracy using trees trained on dog images reached 93% starting from level 10. The average hamming accuracy of all images is calculated in Fig. 4. As it is shown, the average hamming keeps increasing with the increase of the number of tree levels, then it maintains almost the same value starting from depth 6 to 10 and slight performance degradation after 10 indicating the occurrence of overfitting. This is due to using only $9K$ images for in training.

In Fig. 7, the confusion matrix was calculated where pixels of each body-part of the reference image were checked in the resulted image to determine to which labels of other body parts these pixels were assigned. As shown, higher level trees have better accuracy. In Fig. 7f, some of the limbs had the value zero because either the limb is not shown in the current posture or because only small number of pixels were assigned to it, where these pixels are assigned and labelled to be part of the closest biggest body-part.

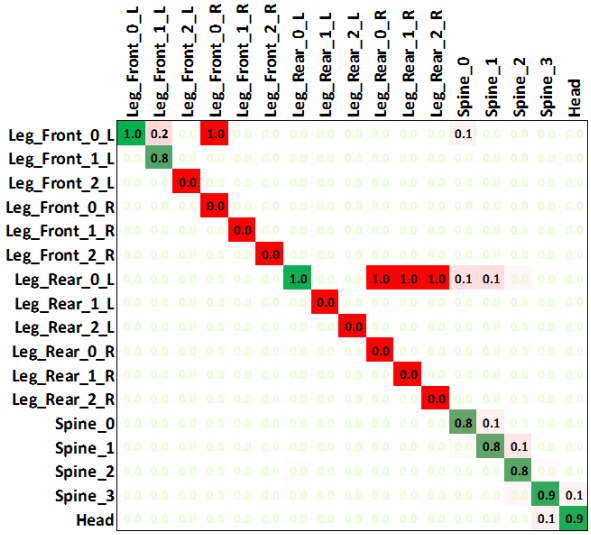
V. Conclusion

In this paper an approach to semantic dog body parts segmentation as an intermediate step towards animals pose estimation using a RGB-D camera is presented. First, a MoCap system is used to capture the movement of the dog. Then, a constructed skeleton is fitted on a dog model. A pixel-wise labelled animal training dataset is generated. The MapReduce component of the Hadoop distributed ecosystem is utilised to train the random decision forest. Results show the successful recognition and identification of the dog body-parts with an accuracy of 93%. This rate can be significantly improved by designing an ensemble of RDFs to recognise different levels

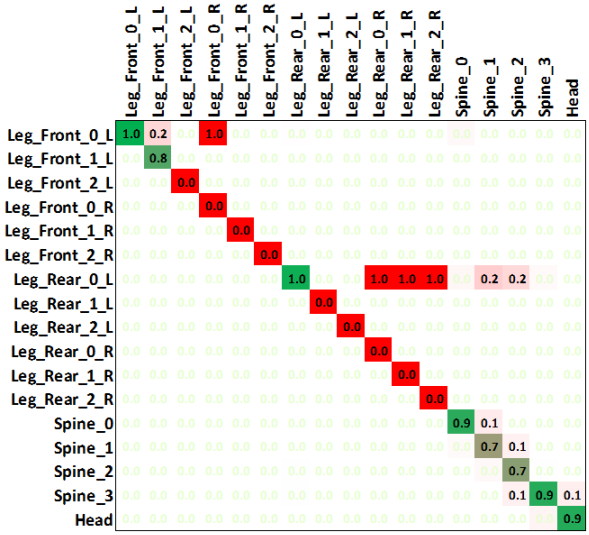


(a) Tree-2

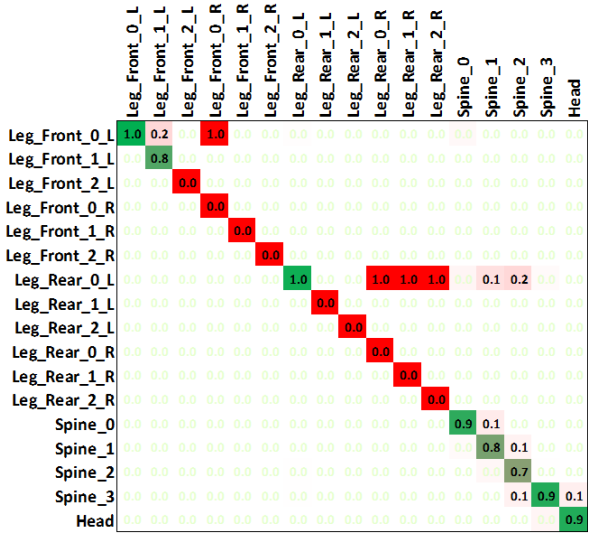
(b) Tree-4



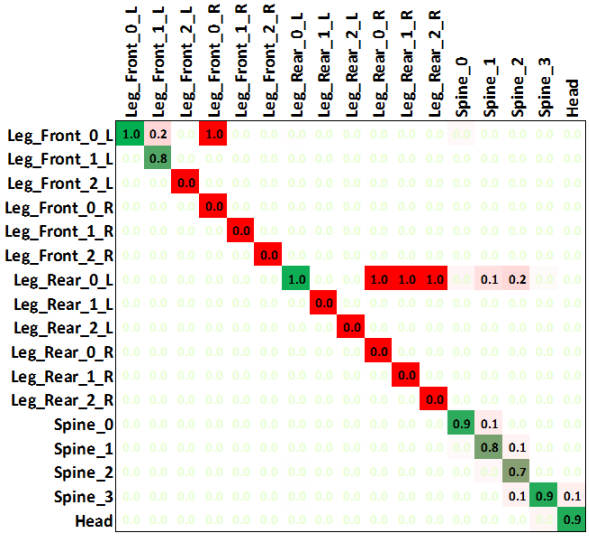
(c) Tree-6



(d) Tree-10



(e) Tree-15



(f) Tree-20

Fig. 7: The confusion matrix representing the percentage of the number of pixels assigned to each dog body-part of the input reference image when using different levels of trained RDF trees.

of details. In order to achieve this, an accustomed fusion algorithm will be designed to fuse indexed label images of the recognised labels from different RDFs as described in [30].

ACKNOWLEDGEMENT

This research was fully supported by the Centre for Intelligent Systems Research (CISR) at Deakin University.

REFERENCES

[1] B. Lange, C. Y. Chang, E. Suma, B. Newman, A. S. Rizzo, and M. Bolas, "Development and evaluation of low cost game-based balance rehabilitation tool using the microsoft kinect sensor," in *IEEE engineering in medicine and biology society*, 2011, pp. 1831–1834.

[2] J. Tong, J. Zhou, L. Liu, Z. Pan, and H. Yan, "Scanning 3d full human bodies using kinects," in *IEEE Transactions on Visualization and Computer Graphics*, 2012, pp. 643–650.

[3] T. Dutta, "Evaluation of the kinect sensor for 3-d kinematic measurement in the workplace," in *Applied Ergonomics*, 2012, pp. 645–649.

[4] H. Haggag, M. Hossny, D. Filippidis, D. Creighton, S. Nahavandi, and V. Puri, "Measuring depth accuracy in rgbd cameras," in *International Conference on Signal Processing and Communication Systems*, 2013.

[5] K. Khoshelham and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," in *Sensors*, 2012, pp. 1437–1454.

[6] H. Haggag, M. Hossny, S. Haggag, S. Nahavandi, and D. Creighton, "Safety applications using kinect technology," in *IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2014, pp. 2164–2169.

[7] M. Hossny, D. Filippidis, W. Abdelrahman, H. Zhou, M. Fielding, J. Mullins, L. Wei, D. Creighton, V. Puri, and S. Nahavandi, "Low cost multimodal facial recognition via kinect sensors," in *The Land Warfare Conference*, 2012, pp. 77–86.

[8] M. Zollhofer, M. Martinek, G. Greiner, M. Stamminger, and J. SuBmuth, "Automatic reconstruction of personalized avatars from 3d face scans," in *Computer Animation and Virtual Worlds*, 2011, pp. 195–202.

[9] L. Wei, V. Le, W. Abdelrahman, M. Hossny, D. Creighton, and S. Nahavandi, "Kinect crowd interaction," in *Asia Pacific Simulation Technology and Training, Adelaide, South Australia*, 2012.

[10] K. F. Li, K. Lothrop, E. Gill, and S. Lau, "A web-based sign language translator using 3d video processing," in *14th International Conference on Network-Based Information Systems*, 2011, pp. 356–361.

[11] M. N. K. Boulos, B. J. Blanchard, C. Walker, J. Montero, A. Tripathy, and R. Gutierrez-Osuna, "Web gis in practice x: a microsoft kinect natural user interface for google earth navigation," in *International Journal of Health Geographics*, 2011, p. 45.

[12] A. D. Wilson, "Using a depth camera as a touch sensor," in *ACM International Conference on Interactive Tabletops and Surfaces*, 2010, pp. 69–72.

[13] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera," in *The 24th annual ACM symposium on user interface software and technology*, 2011, pp. 559–568.

[14] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, "Real-time human pose recognition in parts from single depth images," in *Communications of the ACM*, 2013, pp. 116–124.

[15] J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *IEEE Conference on Computer vision and pattern recognition (CVPR)*, 2008, pp. 1–8.

[16] T. Sharp, "Implementing decision trees and forests on a gpu," in *Computer Vision (ECCV)*. Springer, 2008, pp. 595–608.

[17] J. R. Quinlan, "Induction of decision trees," in *Machine learning*. Springer, 1986, pp. 81–106.

[18] K. Buys, C. Cagniart, A. Baksheev, T. De Laet, J. De Schutter, and C. Pantofaru, "An adaptable system for rgb-d based human body detection and pose estimation," in *Journal of Visual Communication and Image Representation*. Elsevier, 2014, pp. 39–52.

[19] T. Zhang, A. Wiliem, G. Hemsony, and B. C. Lovell, "Detecting kangaroos in the wild: the first step towards automated animal surveillance," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1961–1965.

[20] D. Forslund and J. Bjarkefur, "Night vision animal detection," in *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*. IEEE, 2014, pp. 737–742.

[21] A. Kyme, S. Se, S. Meikle, G. Angelis, W. Ryder, K. Popovic, D. Yatigammana, and R. Fulton, "Markerless motion tracking of awake animals in positron emission tomography," *Medical Imaging, IEEE Transactions on*, vol. 33, no. 11, pp. 2180–2190, 2014.

[22] "Blender 3d modeling software," 1995.

[23] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," in *Communications of the ACM*, 2008, pp. 107–113.

[24] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees." *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 9, pp. 1465–79, 2006.

[25] L. Breiman, "Random forests," in *Machine learning*. Springer, 2001, pp. 5–32.

[26] J. R. Quinlan, "Induction of decision trees," vol. 1, no. 1. Springer, 1986, pp. 81–106.

[27] A. Criminisi, J. Shotton, and E. Konukoglu, *Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning*. Now, 2012.

[28] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, 2010, pp. 1–10.

[29] R. W. Hamming, "Error detecting and error correcting codes," in *Bell System technical journal*, 1950, pp. 147–160.

[30] M. Hossny, S. Nahavandi, and D. Creighton, "Color map-based image fusion," in *IEEE International Conference on Industrial Informatics (INDIN)*, 2008, pp. 52–56.